

【特許請求の範囲】

【請求項1】 リクエスト側からレスポンス側への片方向のリクエストと該リクエストに対する前記レスポンス側から前記リクエスト側へのレスポンスを行うリクエスト／レスポンス型の同期通信のみを用いるシステムにおいて、

前記リクエスト側から前記レスポンス側に向けてメッセージを送信する際は、前記レスポンス側がメッセージを受信したことを、前記リクエスト側に返信するまで、同じメッセージを前記レスポンス側に送信し続け、
前記レスポンス側から前記リクエスト側に向けてメッセージを送信する際は、前記リクエスト側がメッセージの受信要求を行ない、前記リクエスト側がメッセージを受信したことを、新たな通信で前記レスポンス側に受信通知するまで、前記レスポンス側は同じメッセージを前記リクエスト側に送信し続け、
それにより、双方向でメッセージの送受信を可能にした双方向通信手法。

【請求項2】 送信するメッセージ内に一意の識別子を付加し、受信側で重複チェックを行なうことにより、同じメッセージを重複して受信することを防ぐようにした請求項1記載の双方向通信手法。

【請求項3】 前記メッセージの形式にXMLを利用し、前記メッセージとは異なる特定のネームスペースでメッセージ内に前記識別子を付加することで、メッセージの形式を変更せずに、同じメッセージを重複して受信することを防ぐようにした請求項2記載の双方向通信手法。

【請求項4】 前記メッセージの一意性は任意の形で実現し、メッセージの一意性を検証する際に、該メッセージのダイジェスト値を利用することで、同じメッセージを重複して受信することを防ぐようにした請求項1記載の双方向通信手法。

【請求項5】 受信メッセージをバッファに保管完了した後にメッセージを受信したことをメッセージ送信側に通知するようにし、前記バッファの容量不足により前記受信メッセージの保管に失敗した場合は前記バッファ内の古いメッセージを削除することにより受信メッセージを前記バッファに保管するようにした請求項1記載の双方向通信方法。

【請求項6】 前記メッセージの送信側が送信するメッセージに送信メッセージ用電子署名を付加し、該送信メッセージ用電子署名を受信側が保管することによって送信側による送信事実の否認を防止することができるようにすることと、

前記メッセージの受信側が前記メッセージの受信に応答して送信するメッセージ受信通知に、受信通知用電子署名を付加し、該受信通知用電子署名を、送信側が保管することによって受信側による受信事実の否認を防止することができるようにすることと、のすくなくとも一方を

採用した請求項2から4のいずれか一項記載の双方向通信手法。

【請求項7】 前記署名付きの受信メッセージを受信側のバッファに保管完了した後にメッセージを受信したことをメッセージ送信側に通知するようにし、前記バッファの容量不足により前記受信メッセージ及び前記署名の少なくとも一方の保管に失敗した場合は前記バッファ内の古いメッセージを削除した後に前記バッファの容量が依然として不足の場合は前記バッファ内の古い署名を削除することにより前記署名付きの受信メッセージを前記バッファに保管するようにした請求項6記載の双方向通信方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は双方向通信方法に関し、より詳細にはリクエスト側からレスポンス側への片方向のリクエストとそのリクエストに対するレスポンス側からリクエスト側へのレスポンスを行うリクエスト／レスポンス型の同期通信のみを用いるシステムにおいて、双方向でメッセージを送受信すること、信頼性を保証すること、及び送受信の否認を防止することを可能にした双方向通信手法に関する。

【0002】

【従来の技術】図1はリクエスト／レスポンス型の同期通信のみを用いる従来のシステムの構成を示すブロック図である。図において、11はリクエスト側であるクライアント、12はレスポンス側であるサーバ、13はクライアント11の入口に設けられたファイアウォールである。このように、ファイアウォール13が設けられていると、その内部とその外側との通信は、片方向のリクエスト／レスポンス型の同期通信を用いることが多い。

【0003】

【発明が解決しようとする課題】片方向のリクエスト／レスポンス型の同期通信のみを用いた場合、従来のHTTPなどのプロトコルでは、図1に示すようにメッセージ送達とその確認はリクエスト側11からレスポンス側12への1方向しか実現できなかった。サーバ側12からクライアント側11にメッセージを送信しようとしても、ファイアウォール13によりブロックされるので、クライアント側11には到達しないという問題があった。

【0004】また、送受信の記録を相手側に提示するだけでは、確かに送受信をしたという証拠とならないので、送信事実を受信側が否認したり、受信事実を送信側が否認したりすることができてしまうという問題もあった。このような、送信事実の否認、受信事実の否認は、特にインターネットを介して金銭取引を行う場合等において重大である。

【0005】本発明の目的は、上記従来技術における問題に鑑み、片方向のリクエスト／レスポンス型の通信プ

3

ロトコルを用いて、後述する再送手順により双方向の送達確認を可能にする通信方法を提供することにある。

【0006】本発明の他の目的は、上記通信方法において、メッセージに一意の識別子を付加したり、送受信されたメッセージに対して署名を作成し、これを交換することにより、送受信が確に行なわれたことを証拠として残し、それにより、送信事実、受信事実の否認を防止することにある。

【0007】本発明のさらに他の目的は、XML (Extensive Markup Language) を用いてメッセージを構築することにより、メッセージの一意性を保証する処理の自動化を容易にすることにある。

【0008】

【課題を解決するための手段】上記の目的を達成するために、本発明の第1の態様により、レスポンス側からリクエスト側に向けてメッセージを送信する際は、リクエスト側がメッセージの受信要求を行ない、リクエスト側がメッセージを受信したことを、新たな通信でレスポンス側に受信通知するまで、レスポンス側は同じメッセージをリクエスト側に送信し続けることにより、双方向でメッセージの送受信を可能にした双方向通信手法が提供される。

【0009】リクエスト側からの受信要求に応じてであれば、レスポンス側はメッセージをリクエスト側に送信できるので、リクエスト側からとレスポンス側からの双方向からメッセージの送信が可能になる。

【0010】本発明の第2の態様により、上記第1の態様において、送信するメッセージ内に一意の識別子を付加し、受信側で重複チェックを行なうことにより、同じメッセージを重複して受信することを防ぐようにした。

【0011】本発明の第3の態様により、上記第2の態様において、メッセージの形式にXMLを利用し、メッセージとは異なる特定のネームスペースでメッセージ内に識別子を付加することで、メッセージの形式を変更せずに、同じメッセージを重複して受信することを防ぐようにした。

【0012】本発明の第4の態様により、上記第1の態様において、メッセージの一意性は任意の形で実現し、メッセージの一意性を検証する際に、メッセージのダイジェスト値を利用することで、同じメッセージを重複して受信することを防ぐようにした。

【0013】上記第2から第4の態様により、同じメッセージを重複して受信することを防ぐことができるので、通信効率を向上させることができる。

【0014】本発明の第5の態様により、上記第2から第4の態様のいずれかにおいて、メッセージの送信側が送信するメッセージに対して送信メッセージ用電子署名を付加し、この送信メッセージ用署名を受信側が保管することによって送信側による送信事実の否認を防止することができるようにすることと、メッセージの受信側が

4

メッセージの受信に応答して送信するメッセージ受信通知に、受信通知用電子署名を付加し、この受信通知用電子署名を、送信側が保管することによって受信側による受信事実の否認を防止することができるようにすることと、の少なくとも一方を採用する。

【0015】これにより、送信側による送信事実の否認や受信側による受信事実の否認を防止することができる。

【0016】

【発明の実施の形態】以下、本発明の実施の形態を図面を参照しながら説明する。以下の説明で同一参照番号は同一のものを表し、図1に示したファイアウォール13は図面の簡単化のために図示を省略してある。

【0017】図2は、本発明の第1の実施の形態によりリクエスト側11からレスポンス側12にメッセージの送信する場合のデータの流れを示すブロック図である。リクエスト側11の例としては、商品の売主や買主等、サーバを持たないクライアントがある。レスポンス側12の例としては、申し込み書等をデータベースに保管しているセンタの役割を果たすサーバがある。

【0018】図3は図2に示したメッセージを送信する場合の処理の流れを説明するフローチャートである。

【0019】図2および図3で説明するメッセージの送信自体は、従来の片方向通信で行われていたものである。

【0020】図2及び図3において、リクエスト側11では、ステップS31で1つ又は複数のメッセージを作成し、ステップS32でそのメッセージを保管し、ステップS33で送信したいメッセージ21をレスポンス側12に送信する。メッセージ21には受信通知の返信を要求するリクエストが含まれている。

【0021】レスポンス側12では、ステップS34でそのメッセージを受信し、ステップS35で受信したメッセージ中のID（識別子）と同じIDがレスポンス側12に存在するかを検索する。その検索の結果ステップS36で同じIDがレスポンス側12に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップS37にて受信メッセージを保管し、ステップS38で受信通知を作成する。ステップS36の検索の結果、同じIDがレスポンス側12に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップS38にて受信通知を作成する。次いでステップS39にて受信通知22をリクエスト側11に向けて返信する。受信通知22には、リクエストに対するレスポンスである旨の情報が含まれている。

【0022】リクエスト側11ではステップS40にて所定時間間隔毎に上記受信通知22の受信を監視し、ステップS41にて対応するメッセージを送信したか、即ちステップS33にてリクエスト側11から送信された

5

メッセージがレスポンス側12に正常に受信されたかを
受信通知22の有無により判断し、未だ受信通知22を
受信していない場合は対応するメッセージ21を送信し
ていないと判断してステップS33に戻りメッセージ2
1をレスポンス側12に再送する。

【0023】ステップS41の判断でメッセージ21が
送信済みと判定された場合は、ステップS42にてリク
エスト側11に保管されている対応するメッセージ21
を削除する。

【0024】上記図2及び図3により説明した、リク
エスト側11からレスポンス側12側への片方向メッセ
ージの送信自体は従来から可能であった。

【0025】図4は本発明の第1の実施の形態により双
方向通信を可能にする場合のデータの流れを示すブロッ
ク図であり、図5は図4に示したメッセージを送信する
場合の処理の流れを説明するフローチャートである。

【0026】図4及び図5において、レスポンス側12
では、ステップS51で1つ又は複数のメッセージを作
成し、ステップS52でそのメッセージを保管してお
く。

【0027】リクエスト側11では、ステップS53に
て受信要求41をレスポンス側12に送信する。レスポ
ンス側12はステップS54でこの受信要求41を受信
し、ステップS55でその受信要求に応ずるメッセージ
42を保管されているメッセージの中から検索する。受
信要求に応ずるメッセージ42が存在すればステップS
56にてそのメッセージ42をリクエスト側11に送信
する。

【0028】リクエスト側11ではそのメッセージ42
をステップS57で受信し、ステップS58で受信した
メッセージ中のID(識別子)と同じIDがリクエスト
側11に存在するかを検索する。検索した結果、ステッ
プS59で同じIDがリクエスト側11に存在しなければ、
今回受信したメッセージは新たなメッセージなので、
ステップS60にて受信メッセージを保管し、ステッ
プS61で受信通知43を作成する。ステップS59
の検索の結果、同じIDがリクエスト側11に存在すれ
ば、受信済みのメッセージなので、今回受信したメッセ
ージは保管せずに、ステップS61にて受信通知43を
作成する。次いでステップS62にて受信通知43をレ
スポンス側12に向けて返信する。

【0029】レスポンス側12ではステップS63にて
所定時間間隔毎に上記受信通知43の受信を監視し、ス
テップS64にて対応するメッセージを送信したか、即
ちステップS63にてレスポンス側12から返信された
メッセージがリクエスト側11に正常に受信されたかを
受信通知43の有無により判断し、受信通知43が受信
されていればメッセージ42がリクエスト側11からレ
スポンス側12に送信済みと判定され、ステップS65
にてレスポンス側12に保管されている、受信要求41

6

に対応するメッセージ42を削除する。

【0030】未だ受信通知43を受信していない場合は
ステップS66にて処理を終了する。

【0031】リクエスト側11からレスポンス側12に
は受信要求41が定期的に送信されるので、このメッセ
ージ42の送信はリクエスト側11から受信通知43が
到着するまで、定期的に繰り返して行われる。受信通知4
3がリクエスト側11から送られなければ、レスポンス
側12ではメッセージ42が確かにリクエスト側11に
届いたかどうか不明であるが、本発明の実施の形態によ
り、メッセージ42の受信に回答して受信通知43をリ
クエスト側11からレスポンス側12に返信するように
したことにより、レスポンス側12では確実にメッセ
ージ42がリクエスト側に届いたことを認識でき、結果的
に双方向のメッセージのやり取りが可能になる。

【0032】このように、リクエスト側11に設けられ
たファイアウォールでリクエスト側11からのみ外部にメ
ッセージを送ることができるシステムにおいて、レスポ
ンス側12からリクエスト側11にメッセージを送信す
る必要性及びレスポンス側12から送られたメッセージ
がリクエスト側11に確実に受信されたことを認識でき
る必要性は、特に金融に関する通信や企業内のクライ
アントと企業外の別の企業との間の文書の通信や、レス
ポンス側12で発行する何らかの申し込み書等で必要な
る。例えば、申し込み書の配信等のように、レスポンス
側12から送られるウェブブラウザの画面中にメッセ
ージを含ませたいという要求がある。従来ではこの場合、
レスポンス側12からリクエスト側11へのメッセージ
の配信は不可能であったが、本発明の実施の形態によれ
ば上記のように確実にメッセージをレスポンス側12か
らリクエスト側11に配信することが可能になる。

【0033】尚、メッセージの送受信にはSOAP(Si
mple Object Access Protocol)等のプロトコルを用い
て実現できる。

【0034】図6は本発明の第2の実施の形態によりメ
ッセージ中の定まったフォーマットの中の特定の場所に
メッセージの識別子(ID)を埋め込む方法を説明する
図である。図において、ルートタグ<Order>からルー
トタグ</Order>までが定まったフォーマットの注文書
のメッセージであり、このメッセージ中のmessageIdと
いう2つのタグの間にIDとして"12345678"が埋め込ま
れている。受信側ではこのメッセージのフォーマットを
知っているので受信メッセージ中のIDを自分が持つID
と照合することにより受信済みメッセージかどうかを
判断する。図示例はXMLで記載された注文書のメッセ
ージを例としてあげてあるが、定まったフォーマットの
メッセージであればどのような言語で記載されていても
よい。これにより、受信側ではプログラムが解釈可能な
フォーマットからメッセージIDを検索可能になる。

【0035】図7は本発明の第3の実施の形態によりメ

7

ッセージ内にメッセージ内容とは別にネームスペースでメッセージIDを埋め込む方法を説明する図である。図において、ルートタグ<Order>からルートタグ</Order>までは図6と同じく注文書のメッセージであるが、このメッセージ中のプレフィックスssrs:とその後の属性名messageIdの後に="12345678"というメッセージIDを埋め込んである。これにより、受信側ではプログラムによりプレフィックスの属性を探せば、IDを見つけることができ、メッセージの一意性を検証することができる。

【0036】図7の例では、ルートタグとして<Order>を用いているが、プレフィックスとその属性ssrs:messageIdの後にメッセージIDを埋め込めばよいので、ルートタグが何であってもIDを見つけることができる。また、メッセージのフォーマットが何であってもよい。

【0037】図8は本発明の第4の実施の形態により、メッセージのダイジェストを利用してメッセージの一意性を検証する方法を説明する図である。この場合は、ルートタグ<Order>からルートタグ</Order>までのメッセージの中の任意の場所に<orderId>order012345678</orderId>というIDを含む記号を挿入した後に、SHA1、MD5等のアルゴリズムに従ってこのメッセージをダイジェストに圧縮して送信する。受信側でこのダイジェストを所定のアルゴリズムを用いて計算する。受信メッセージの内容及びIDのいずれかが、既に受信したメッセージの内容及びIDのいずれかと異なれば、ダイジェストの計算値も異なる、受信メッセージの内容及びIDのいずれもが、既に受信したメッセージの内容及びIDのいずれもと一致していれば、ダイジェストの計算値も一致する。これによっても、受信メッセージの一意性の検証が可能になる。図8においても、メッセージは注文書に限定されずどのような内容でもよい。また、メッセージのフォーマットも任意でよい。

【0038】図9は本発明の第5の実施の形態により送信者による送信の事実を後に送信者が否認することを防止する方法におけるデータの流れを説明するブロック図であり、図10は図9に示したシステムにおいて、リクエスト側11からレスポンス側12にメッセージを送信する場合において、送信事実の送信者による否認を防止する処理の流れを説明するフローチャートである。

【0039】図10と図3との相違点は、図10においてはリクエスト側11でステップS102にて送信署名を作成し、レスポンス側12でステップS106及びステップS107で送信署名の検証をし、ステップS111でメッセージの外に署名も保管し、リクエスト側11でステップS115にて送信署名エラーが返信されたかを判断することが追加されていることである。

【0040】より詳細には、図9及び図10において、リクエスト側11では、ステップS101で1つ又は複数のメッセージを作成し、ステップS102で送信者だ

8

けが作成できる送信者署名を作成し、ステップS103でその署名をメッセージとともに保管し、ステップS104で送信したいメッセージ21aをレスポンス側12に送信する。メッセージ21には受信通知の返信を要求するリクエストと署名とが含まれている。

【0041】レスポンス側12では、ステップS105でそのメッセージを受信し、ステップS106で送信署名を送信者の公開鍵で検証する。ステップS107でその検証結果が不正な署名であれば、ステップS108に進んで署名検証エラーを受信通知22に載せてリクエスト側11に返信する。ステップS107で署名の検証結果が正しければ、ステップS109にて図6から図8のいずれかの方法により、受信したメッセージ中のID(識別子)と同じIDがレスポンス側12に存在するかを検索する。その検索の結果ステップS110にて同じIDがレスポンス側12に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップS111にて受信メッセージと共に送信者の署名90をデータベース91に保管し、ステップS112で受信通知を作成する。ステップS110の検索の結果、受信メッセージがレスポンス側12に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップS112にて受信通知を作成する。次いでステップS113にて受信通知22をリクエスト側11に向けて返信する。受信通知22には、リクエストに対するレスポンスである旨の情報が含まれている。

【0042】リクエスト側11ではステップS114にて所定時間間隔毎に上記受信通知22の受信を監視し、ステップS115にて送信署名エラーが返信されたかを判定し、送信署名エラーが返信された場合はステップS116にてエラー処理をする。

【0043】ステップS115の判断で送信署名エラーでないと判定されると、ステップS117にて対応するメッセージを送信したかに正常に受信されたか、即ちステップS104にてリクエスト側11から送信されたメッセージがレスポンス側12を受信通知22の有無により判断し、未だ受信通知22を受信していない場合は対応するメッセージ21aを送信していないと判断してステップS104に戻りメッセージ21aをレスポンス側12に再送する。

【0044】ステップS117の判断でメッセージ21aが送信済みと判定された場合は、ステップS118にてリクエスト側11に保管されている対応するメッセージ21を削除する。

【0045】これにより、リクエスト側11が送信した事実を否認しても、レスポンス側12のデータベース91にはリクエスト側11でのみ作成可能な署名が格納されているので、リクエスト側11による送信事実の否認をレスポンス側12は拒否できる。

【0046】図11は、図9に示したシステムにおい

9

て、レスポンス側12からリクエスト側11にメッセージを送信する場合において、送信事実の送信者による否認を防止する処理の流れを説明するフローチャートである。

【0047】図11と図5との相違点は、図11においては、レスポンス側12でステップS122にて送信署名を作成し、リクエスト側11ではステップS129、ステップS130及びステップS131で送信署名検証を行い、ステップS134でメッセージの外に署名も保管し、レスポンス側12ではステップS138及びステップS139で送信署名エラーかを判断し処理することが追加されていることである。

【0048】より詳細には、図9及び図11において、レスポンス側12では、ステップS121で1つ又は複数のメッセージを作成し、ステップS122で送信者のみが作成できる送信書名を作成する。次いでステップS123でその送信署名をメッセージとともに保管しておく。

【0049】リクエスト側11では、ステップS124にて受信要求41をレスポンス側12に送信する。レスポンス側12はステップS125でこの受信要求41を受信し、ステップS126でその受信要求に応ずるメッセージ42を保管されているメッセージの中から検索する。受信要求に応ずるメッセージ42aが存在すればステップS127にてそのメッセージ42aに署名92を含ませてリクエスト側11に送信する。

【0050】リクエスト側11ではそのメッセージ42aをステップS128で受信し、ステップS129で受信したメッセージ中の送信署名が正しいかどうかを送信者の公開鍵で検証する。正しくなければステップS131で署名検証エラーをレスポンス側12に返信する。正しければステップS132及びステップS133にて受信メッセージ中のID（識別子）と同じIDがリクエスト側11に存在するかを検索する。検索した結果、同じIDがレスポンス側12に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップS134にて受信メッセージを署名92とともにデータベース93に保管し、ステップS135で受信通知43を作成する。ステップS133の検索の結果、同じIDがリクエスト側11に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップS135にて受信通知43を作成する。次いでステップS136にてその受信通知をレスポンス側12に向けて返信する。

【0051】レスポンス側12ではステップS137にて所定時間間隔毎に上記受信通知43の受信を監視し、ステップS138にて送信署名がエラーかを判断する。エラーであれば、ステップS139にてエラー処理をする。エラーでなければステップS140にて対応するメッセージを送信したか、即ちステップS127にてレス

10

ポンス側12から返信されたメッセージがリクエスト側11に正常に受信されたかを受信通知43の有無により判断し、メッセージ42aが送信済みと判定された場合は、ステップS141にてレスポンス側12に保管されている対応するメッセージ42aを削除する。

【0052】未だ受信通知43を受信していない場合はステップS142にて処理を終了する。

【0053】これにより、レスポンス側12が送信した事実を否認しても、リクエスト側11のデータベース92にはレスポンス側12でのみ作成可能な署名92が格納されているので、レスポンス側12による送信事実の否認をリクエスト側11は拒否できる。

【0054】図12は本発明の第6の実施の形態により受信者による受信の事実を後に受信者が否認することを防止する方法におけるデータの流れを説明するブロック図であり、図13は図12に示したシステムにおいて、リクエスト側11からレスポンス側12にメッセージを送信する場合において、受信事実の受信者による否認を防止する処理の流れを説明するフローチャートである。

【0055】図13と図3との相違点は、図13においてはレスポンス側12でステップS158にて受信署名を作成し、リクエスト側11でステップS163及びステップS164で受信署名の検証をし、ステップS165で受信署名を保管することが追加されていることである。

【0056】より詳細には、図12及び図13において、リクエスト側11では、ステップS151で1つ又は複数のメッセージを作成し、ステップS152でメッセージを保管し、ステップS153で送信したいメッセージ21をレスポンス側12に送信する。メッセージ21には受信通知の返信を要求するリクエストが含まれている。

【0057】レスポンス側12では、ステップS154でそのメッセージを受信し、ステップS155で図6から図8のいずれかの方法により、受信したメッセージ中のID（識別子）と同じIDがレスポンス側12に存在するかを検証する。その検証の結果ステップS156にて同じIDがレスポンス側12に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップS157にて受信メッセージを保管し、ステップS158で受信者だけが作成できる受信署名120を作成する。ステップS156の判断の結果、同じIDがレスポンス側12に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップS159にて受信通知を作成する。次いでステップS160にて受信通知22aをリクエスト側11に向けて返信する。受信通知22aには、リクエストに対するレスポンスである旨の情報に加えて受信署名120も含まれている。

【0058】リクエスト側11ではステップS161に

11

て所定時間間隔毎に上記受信通知22の受信を監視し、ステップS162にて対応するメッセージを送信したか、即ち、ステップS153にてリクエスト側11から送信したメッセージがレスポンス側に正常に受信されたかを受信通知22aの有無により判断し、未だ受信通知22aを受信していない場合は対応するメッセージ21を送信していないと判断してステップS153に戻りメッセージ21をレスポンス側12に再送する。

【0059】ステップS164の判断でメッセージ21が送信済みと判定された場合は、ステップS165にて受信署名をデータベース121に補完し、ステップS166にてリクエスト側11に保管されている対応するメッセージ21を削除する。

【0060】これにより、レスポンス側12が受信した事実を否認しても、リクエスト側11のデータベース121にはレスポンス側12でのみ作成可能な署名が格納されているので、レスポンス側12による受信事実の否認をリクエスト側11は拒否できる。

【0061】図14は、図12に示したシステムにおいて、レスポンス側12からリクエスト側11にメッセージを送信する場合において、受信事実の受信者による否認を防止する処理の流れを説明するフローチャートである。

【0062】図14と図5との相違点は、図14においては、リクエスト側11でステップS181にて受信署名を作成し、レスポンス側12ではステップS186及びステップS187で送信署名検証を行い、ステップS188で受信署名を保管することである。

【0063】より詳細には、図12及び図14において、レスポンス側12では、ステップS171で1つ又は複数のメッセージを作成し、ステップS172でそのメッセージを保管しておく。

【0064】リクエスト側11では、ステップS173にて受信要求41をレスポンス側12に送信する。レスポンス側12はステップS174でこの受信要求41を受信し、ステップS175でその受信要求に応ずるメッセージ42を保管されているメッセージの中から検索する。受信要求に応ずるメッセージ42が存在すればステップS176にてそのメッセージ42をリクエスト側11に送信する。

【0065】リクエスト側11ではそのメッセージ42をステップS177で受信し、ステップS178で受信したメッセージ中のID（識別子）と同じIDがリクエスト側11に存在するかを検証する。検証した結果、ステップS179で同じIDがレスポンス側12に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップS180にて受信メッセージを保管し、ステップS181で受信者のみが作成できる受信書名122を作成し、ステップS184で受信署名122を含む受信通知43aを作成する。ステップS179の

12

検索の結果、同じIDがリクエスト側11に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップS181で受信者のみが作成できる受信書名122を作成し、ステップS184で受信署名122を含む受信通知43aを作成する。次いでステップS183にてその受信通知43aをレスポンス側12に向けて返信する。

【0066】レスポンス側12ではステップS184にて所定時間間隔毎に上記受信通知43aの受信を監視し、ステップS185にて対応するメッセージを送信したか、即ち、ステップS176にてレスポンス側12から送信されたメッセージがリクエスト側11に正常に受信されたかを受信通知43aの有無により判断し、メッセージ42が送信済みと判定された場合は、ステップS186にて受信署名を受信者の公開鍵で検証する。ステップS187の判断で正しい受信署名であると判定されると、ステップS188にて受信署名をデータベース123に保管し、ステップS189にてレスポンス側12に保管されている対応するメッセージ42を削除する。

【0067】ステップS185の判断で未だ受信通知43を受信していない場合、又はステップS187の判断で正しい受信署名ではないと判定された場合はステップS190にて処理を終了する。

【0068】これにより、リクエスト側11が受信した事実を否認しても、レスポンス側12のデータベース123にはリクエスト側11でのみ作成可能な受信署名132が格納されているので、リクエスト側11による受信事実の否認をレスポンス側12は拒否できる。

【0069】図15は本発明の第7の実施の形態により送受信者による送受信の事実を後に送受信者が否認することを防止する方法におけるデータの流れを説明するブロック図であり、図16は図15に示したシステムにおいて、リクエスト側11からレスポンス側12にメッセージを送信する場合の処理の流れを説明するフローチャートである。

【0070】図16は図10と図13とを融合したものである。

【0071】より詳細には、図15及び図16において、リクエスト側11では、ステップS191で1つ又は複数のメッセージを作成し、ステップS192で送信者だけが作成できる送信署名を作成し、ステップS193でその送信署名をメッセージとともにを保管し、ステップS194で送信したいメッセージ21aをレスポンス側12に送信する。メッセージ21aには受信通知の返信を要求するリクエストと送信署名とが含まれている。

【0072】レスポンス側12では、ステップS195でそのメッセージを受信し、ステップS196で送信署名を送信者の公開鍵で検証する。ステップS197でその検証結果が不正な署名であれば、ステップS198に

13

進んで署名検証エラーを受信通知22aに載せてリクエスト側11に返信する。ステップS197で署名の検証結果が正しければ、ステップS199にて図6から図8のいずれかの方法により、受信したメッセージ中のID（識別子）と同じIDがレスポンス側12に存在するかを検索する。その検索の結果ステップS200にて同じIDがレスポンス側12に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップS201にて受信メッセージと共に送信者の署名90をデータベース91に保管し、ステップS202で受信者のみ

10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50

【0073】リクエスト側11ではステップS205にて所定時間間隔毎に上記受信通知22aの受信を監視し、ステップS206にて送信署名エラーが返信されたかを判定し、送信署名エラーが返信された場合はステップS207にてエラー処理をする。

【0074】ステップS206の判断で送信署名エラーでないと判定されると、ステップS208にて対応するメッセージを送信したか、即ち、ステップS194にてリクエスト側11から送信したメッセージがレスポンス側12に正常に受信されたかを受信通知22aの有無により判断し、未だ受信通知22aを受信していない場合は対応するメッセージ21aを送信していないと判断してステップS194に戻りメッセージ21aをレスポンス側12に再送する。

【0075】ステップS206の判断でメッセージ21aが送信済みと判定された場合は、ステップS209にて受信通知22a内の受信署名を受信者の公開鍵で検証する。この検証の結果ステップS210で正しくない受信署名であると判定されると、リクエスト側11からレスポンス側12にメッセージ21aが未だ送信されていないものと判断して、ステップS194に戻りメッセージ21aをレスポンス側12に再送する。

【0076】ステップS210の検証で正しい受信署名であると判定されると、ステップS211にて受信署名をデータベース121に保管し、次いでステップS212にてリクエスト側11に保管されている対応するメッセージ21を削除する。

【0077】これにより、リクエスト側11が送信した事実を否認しても、レスポンス側12のデータベース91にはリクエスト側11でのみ作成可能な送信署名が格納されているので、リクエスト側11による送信事実の否認をレスポンス側12は拒否できるとともに、レスポンス側12が受信した事実を否認しても、リクエスト側11のデータベース121にはレスポンス側12でのみ作成可能な受信署名が格納されているので、レスポンス

14

側12による受信事実の否認をリクエスト側11は拒否できる。

【0078】図17は、図15に示したシステムにおいて、レスポンス側12からリクエスト側11にメッセージを送信する場合において、送受信事実の送受信者による否認を防止する処理の流れを説明するフローチャートである。

【0079】図17は図11と図11と図14を融合したものである。

【0080】より詳細には、図15及び図17において、レスポンス側12では、ステップS220で1つ又は複数のメッセージを作成し、ステップS221で送信者のみで作成できる送信署名を作成する。次いでステップS222でその送信署名をメッセージとともに保管しておく。

【0081】リクエスト側11では、ステップS223にて受信要求41をレスポンス側12に送信する。レスポンス側12はステップS224でこの受信要求41を受信し、ステップS225でその要求に応ずるメッセージ42aを保管されているメッセージの中から検索する。受信要求に応ずるメッセージ42aが存在すればステップS226にてそのメッセージ42aに署名92を含ませてリクエスト側11に送信する。

【0082】リクエスト側11ではそのメッセージ42aをステップS227で受信し、ステップS228で受信したメッセージ中の送信署名が正しいかどうかを送信者の公開鍵で検証する。正しくなければステップS230で署名検証エラーをレスポンス側12に返信する。正しければステップS231及びステップS232にてメッセージ42a中のID（識別子）と同じIDがリクエスト側11に存在するかを検索する。検索した結果、同じIDがレスポンス側12に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップS233にて受信メッセージを署名92とともにデータベース93に保管し、ステップS234で受信者のみで作成できる受信署名122を作成し、ステップS235で受信署名122を含む受信通知43aを作成する。次いでステップS236で受信通知43aをレスポンス側12に送信する。

【0083】ステップS232の検索の結果、同じIDがリクエスト側11に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップS235にて受信署名を作成する。

【0084】レスポンス側12ではステップS237にて所定時間間隔毎に上記受信通知43aの受信を監視し、ステップS238にて送信署名がエラーかを判断する。エラーであれば、ステップS239にてエラー処理をする。エラーでなければステップS240にて対応するメッセージを送信したか、即ち、ステップS226にてレスポンス側12から送信したメッセージがリクエ

15

ト側11に正常に受信されたかを受信通知43aの有無により判断し、メッセージ42aが送信済みと判定された場合は、ステップS241にて受信者の公開鍵により受信署名を検証し、ステップS242で正しい署名であればステップS243で受信署名122をデータベース151に保管し、ステップS244でレスポンス側12に保管されている対応するメッセージ42aを削除する。

【0085】ステップS240にて未だ対応するメッセージを送信していないと判定された場合又はステップS242にて正しい受信署名ではないと判定された場合は、ステップS245にて処理を終了する。

【0086】これにより、レスポンス側12が送信した事実を否認しても、リクエスト側11のデータベース93にはレスポンス側12でのみ作成可能な送信署名が格納されているので、レスポンス側12による送信事実の否認をリクエスト側11は拒否できるとともに、リクエスト側11が受信した事実を否認しても、レスポンス側12のデータベース151にはリクエスト側11でのみ作成可能な受信署名が格納されているので、リクエスト側11による受信事実の否認をレスポンス側12は拒否できる。以上に記載した各実施の形態において、受信したメッセージ及び署名の少なくとも一方の保管が正常に完了しない場合がある。その原因としては、メッセージや署名を保管するバッファの領域が不足している場合や、そのバッファに書き込み禁止フラグが立てられている等によりそのバッファへの書き込み権がない場合や、保存先を間違えた場合や、保存先がデータベースの場合でそのデータベースが起動していない場合等がある。バッファの領域が不足しているためにメッセージや署名の保管ができなかった場合は、以下に記載する本発明の第8及び第9の実施の形態により保管可能になる場合がある。図18は、図3におけるステップS37又は図5におけるステップS60におけるメッセージの保管が成功しなかった場合の本発明の第8の実施の形態によるメッセージ保管方法を説明するフローチャートである。同図において、ステップS250にて署名がない受信メッセージの保管動作を行う。このステップは図3におけるステップS37又は図5におけるステップS60の動作である。次いでステップS251にてメッセージ保存エラーかどうかを判定し、エラーであればステップS252に進んでエラーの原因がメッセージ保存用バッファの領域不足かどうかを判定する。領域不足であればステップS253にてそのメッセージ保存領域の中の最も古いメッセージから順に少なくとも一つのメッセージを削除し、ステップS254にて再度領域不足かを判定する。この判定で依然として領域不足であれば、メッセージ保存に失敗したことを示すメッセージ保存エラーをメッセージの送信側に返信する。ステップS254にて領域不足でなければステップS250に戻りメッセージを保管

16

する。ステップS251にてメッセージ保存エラーでなければステップS255にて受信通知を作成する。ステップS255は図3におけるステップS38又は図5におけるステップS61に相当する。図18に示したメッセージ保管エラーの救済方法は、図13におけるステップS157又は図14におけるステップS180のメッセージ保管が出来なかった場合にも同様に適用できる。また、メッセージに替えて受信した署名の保管が出来なかった場合にも同様に適用できる。即ち、図13のステップS165、図14のステップS188、図16のステップS211、図17のステップS243において受信署名の保管に失敗した場合に、図18に示したフローチャートにおける「メッセージ」を「署名」に変更したフローチャート(図示省略)により、署名保管のエラーを救済できる場合がある。図19は、図16におけるステップS201又は図17におけるステップS233におけるメッセージ及び署名の保管が成功しなかった場合の本発明の第9の実施の形態によるメッセージ及び署名の保管方法を説明するフローチャートである。同図において、ステップS260にて受信メッセージ及び署名の保管動作を行う。このステップは図16におけるステップS201又は図17におけるステップS233の動作である。次いでステップS261にてメッセージ及び署名の少なくとも一方の保存エラーかどうかを判定し、エラーであればステップS262に進んでエラーの原因がメッセージ及び署名の保存領域の領域不足かどうかを判定する。領域不足であればステップS263にてそのメッセージ及び署名の保存領域の中の最も古いメッセージから順に少なくとも一つのメッセージを削除し、ステップS264にて再度領域不足かを判定する。この判定で依然として領域不足であれば、ステップS265にてそのメッセージ及び署名の保存領域の中の最も古い署名を削除し、ステップS266にて再度領域不足かを判定する。この判定で依然として領域不足であれば、署名検証エラーであることを示すメッセージを送信側に返信する。ステップS266にて領域不足でなければステップS260に戻りメッセージ及び署名を保管する。ステップS261にてメッセージ又は署名の保存エラーでなければステップS268受信署名を作成し、ステップS269にて受信通知を作成する。ステップS268及びステップS269は図16におけるステップS202及びステップS203又は図17におけるステップS234及びステップS235に相当する。図19に示した署名付きメッセージ保管エラーの救済方法は、図10におけるステップS111、図11におけるステップS134、図16におけるステップS201にも同様に適用できる。上記のように、バッファの容量不足により受信したメッセージや署名の保管に失敗した場合にも、古いメッセージや署名をバッファから削除することにより確実に受信メッセージや署名を保存できる。以上に記載した

17

各実施の形態において用いられるメッセージの形式の一例を以下に記載する。以下の例ではメッセージはアプリケーション間通信の取り決めを定義したExtensible Markup Language (XML) 形式によりSOAP の標準に従って作成されている。図20はリクエスト側からレスポンス側に対してメッセージの送信要求をする場合のメッセージの内容の一例である。このメッセージではヘッダ内にメッセージタイプとして送信要求を示すRequestを記述する。このメッセージは図5のステップS53、図11のステップS124、図14のステップS173又は図17のステップS223で使用される。図20において、`<SOAP:Envelope>` と `</SOAP:ENVELOPE>` との間の情報の中で、最初の行に `xmlns:SOAP=...` と記載されているのは、このメッセージが SOAP を標準とする XML 形式で記載されたものであることを示している。`<SOAP:Header>` と `</SOAP:Header>` との間の `<rm:Message Header...>` と `</rm:Message Header>` の間がこのメッセージのヘッダである。メッセージヘッダには、メッセージの送信元と、送信先と、メッセージのサービスの種類と、メッセージの種類とが記載されている。この例では送信元は `<rm:From>` で始まる行に記載されているように `requester@anyuri.com` となっておりリクエスト側である。また、送信先は `<rm:To>` で始まる行に記載されているように `responder@someuri.com` となっておりレスポンス側である。また、サービスの種類は `<rm:Service>` で始まる行に `Item Quote Service` と記載されているように、問い合わせのサービスである。さらにメッセージの種類は `<rm:Message Type>` で始まる行に `Request` と記載されているように要求メッセージである。このメッセージの送信要求にはメッセージ本体の情報がないので、`<SOAP:Body>` と `</SOAP:Body>` との間には何も記載されていない。図21は署名無し送信メッセージの一例を示す図である。このメッセージは図3のステップS33におけるリクエスト側からのメッセージ送信に使用される。同図において、`<SOAP:Envelope>` と `</SOAP:ENVELOPE>` との間の情報の中で、最初の行は図20と同じである。また、`<rm:Message Header...>` と `</rm:Message Header>` との間のヘッダにおける、送信元、送信先、サービスの種類も図20の例と同じである。この他にメッセージ種類、メッセージID、送信日時等が記載されている。メッセージの種類は `<rm:Message Type>` で始まる行に `Message` と記載されているようにメッセージである。また、メッセージIDとして `<rm:Message Id>` の行に `20020907-045261-0450@anyuri.com` が記載されている。さらに、送信日時として `<rm:Timestamp>` の行に `2002-09-07T10:19:07` と記載されている。`<rm:Reliable Message...>` と `</rm:Reliable Message>` の間のヘッダには、送信メッセージの信頼性を確保するための情報が記載されている。即ち、`<rm:AckRequested SOAP:must understand...>` は確認メッセージの形式が SOAP の標準に準拠すべきことを記述したものであ

18

る。また、`<rm:Duplicate Elimination/>` は重複受信をしたメッセージを削除するものである。`<SOAP:BODY>` と `</SOAP:BODY>` との間にはアプリケーションに依存する情報本体の内容が記載される。図5におけるステップS56で送信されるメッセージも、送信元と送信先が図22に示したものと逆になるだけで図22と同様の形式で作成される。図22は署名無し受信確認メッセージの一例を示す図である。このメッセージは図3のステップS39で使用される。メッセージの形式は図21に示したものと同様であるので詳細な説明は省略する。サービスの種類が `Item Filing Service` となっていてファイルスルサービスであること、及びメッセージの種類が `Acknowledgement` となっていて確認のメッセージであることに着目される。本例では送信元がレスポンス側であり送信先がリクエスト側であるが、送信元をリクエスト側にし送信先をレスポンス側にすれば図5のステップS62で使用するメッセージとなる。図23から図25は署名有りの送信メッセージの一例を示す図である。このメッセージは図10のステップS104で使用される。このメッセージにおいて、第1行の `<SOAP:Envelope>` から第19行の `</rm:Reliable Message>` までは図21に示した署名無し送信メッセージと同じである。本例では、その次の行に `<wsse:Security xmlns:wsse="...">` から `</wsse:BinarySecurityToken>` までのタグが存在する。ここでは、ウェブ・サービス・セキュリティ (Web Service Security) (wsse) によりセキュリティを確保している。次のタグである `<ds:Signature xmlns="...">` から `</ds:SignatureValue>` までの間で、署名がXML形式で記載されている。署名の対象は、`<ds:Xpath>` のタグに記載されているように、メッセージヘッダと、リライアブルメッセージ (Reliable Message) と、情報本体 (SOAP Body) である。これにより、メッセージヘッダと、リライアブルメッセージと、情報本体に対する改竄を防止できる。このように、送信メッセージに WS-Security 形式の署名を付けることにより、送信者の否認を防止できる。本例においても、送信元をレスポンス側とし送信先をリクエスト側とすれば、図11のステップS126で使用するメッセージとなる。図26から図28は署名有りの受信確認メッセージの一例を示す図である。このメッセージは図10のステップS113、図13のステップS160、又は図16のステップS204で使用される。メッセージの形式は図23から図25に示したものと同様であるので詳細な説明は省略する。サービスの種類が `Item Filing Service` となっていてファイルスルサービスであること、及びメッセージの種類が `Acknowledgement` となっていて確認のメッセージであることに着目される。この例においても、受信確認メッセージに WS-Security 形式の署名を付けることにより、メッセージヘッダと、リライアブルメッセージと、情報本体に対する改竄を防止できる。本例においても、送信元をリクエスト側とし、

送信先をレスポンス側とすることにより図11のステップS136、図14のステップS183、又は図17のステップS236で使用するメッセージとなる。

【0087】

【発明の効果】上記の通り、本発明によれば、片方向のリクエスト／レスポンス型の通信プロトコルを用いて、双方向の送達確認を実現することができるという効果が得られる。

【0088】また、送受信されたメッセージに対して署名を作成し、これを交換することで、送受信が確かに行なわれたことを証拠として残し、送信事実、受信事実の否認を防止することができるという効果が得られる。

【0089】さらに、XMLを用いてメッセージを構築することにより、メッセージの一意性を保証する処理の自動化を容易にすることができるという効果が得られる。さらに、受信メッセージの保存がバッファの容量不足によりエラーとなった場合でも、古いメッセージや古い署名を削除することにより、エラーを回避できるという利点もある。

【図面の簡単な説明】

【図1】リクエスト／レスポンス型の同期通信のみを用いる従来のシステムの構成を示すブロック図である。

【図2】本発明の第1の実施の形態によりリクエスト側からレスポンス側にメッセージの送信する場合のデータの流れを示すブロック図である。

【図3】図2に示したメッセージを送信する場合の処理の流れを説明するフローチャートである。

【図4】本発明の第1の実施の形態により双方向通信を可能にする場合のデータの流れを示すブロック図である。

【図5】図4に示したメッセージを送信する場合の処理の流れを説明するフローチャートである。

【図6】本発明の第2の実施の形態によりメッセージ中の定まったフォーマットの中の特定の場所にメッセージの識別子(ID)を埋め込む方法を説明する図である。

【図7】本発明の第3の実施の形態によりメッセージ内にメッセージ内容とは別にネームスペースでメッセージIDを埋め込む方法を説明する図である。

【図8】本発明の第4の実施の形態により、メッセージのダイジェストを利用してメッセージの一意性を検証する方法を説明する図である。

【図9】本発明の第5の実施の形態により送信者による送信の事実を後に送信者が否認することを防止する方法におけるデータの流れを説明するブロック図である。

【図10】図9に示したシステムにおいて、リクエスト側11からレスポンス側12にメッセージを送信する場合において、送信事実の送信者による否認を防止する処理の流れを説明するフローチャートである。

【図11】図9に示したシステムにおいて、レスポンス側12からリクエスト側11にメッセージを送信する場合

において、送信事実の送信者による否認を防止する処理の流れを説明するフローチャートである。

【図12】本発明の第6の実施の形態により受信者による受信の事実を後に受信者が否認することを防止する方法におけるデータの流れを説明するブロック図である。

【図13】図12に示したシステムにおいて、リクエスト側11からレスポンス側12にメッセージを送信する場合において、受信事実の受信者による否認を防止する処理の流れを説明するフローチャートである。

【図14】図12に示したシステムにおいて、レスポンス側12からリクエスト側11にメッセージを送信する場合において、受信事実の受信者による否認を防止する処理の流れを説明するフローチャートである。

【図15】本発明の第7の実施の形態により送受信者による送受信の事実を後に送受信者が否認することを防止する方法におけるデータの流れを説明するブロック図である。

【図16】図15に示したシステムにおいて、リクエスト側11からレスポンス側12にメッセージを送信する場合の処理の流れを説明するフローチャートである。

【図17】図15に示したシステムにおいて、レスポンス側12からリクエスト側11にメッセージを送信する場合において、送受信事実の送受信者による否認を防止する処理の流れを説明するフローチャートである。

【図18】図3におけるステップS37又は図5におけるステップS60におけるメッセージの保管が成功しなかった場合の本発明の第8の実施の形態によるメッセージ保管方法を説明するフローチャートである。

【図19】図16におけるステップS201又は図17におけるステップS233におけるメッセージ及び署名の保管が成功しなかった場合の本発明の第9の実施の形態によるメッセージ及び署名の保管方法を説明するフローチャートである。

【図20】リクエスト側からレスポンス側に対してメッセージの送信要求をする場合のメッセージの内容の一例である。

【図21】署名無し送信メッセージの一例を示す図である。

【図22】署名無し受信確認メッセージの一例を示す図である。

【図23】署名有りの送信メッセージの一例の一部を示す図である。

【図24】署名有りの送信メッセージの一例の他の一部を示す図である。

【図25】署名有りの送信メッセージの一例の更に他の一部を示す図である。

【図26】署名有りの受信確認メッセージの一例の一部を示す図である。

【図27】署名有りの受信確認メッセージの一例の他の一部を示す図である。

【図28】署名有りの受信確認メッセージの一例の更に他の一部を示す図である。

【符号の説明】

11…リクエスト側
12…レスポンス側
13…ファイアウォール
21…メッセージ
21a…メッセージ
22…受信通知
22a…受信通知
41…受信要求
42…メッセージ

42a…メッセージ
43…受信通知
43a…受信通知
90…署名
91…データベース
92…署名
93…データベース
120…署名
121…データベース
122…署名
151…データベース

【図1】

【図2】

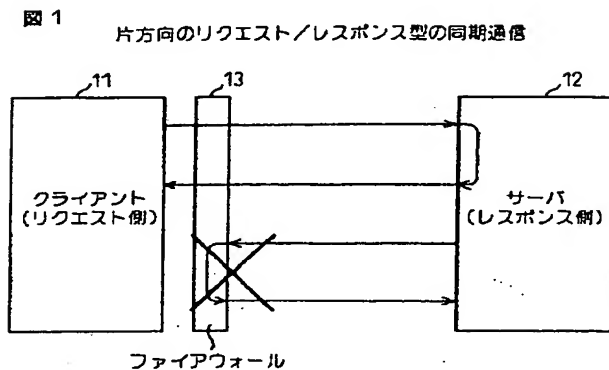
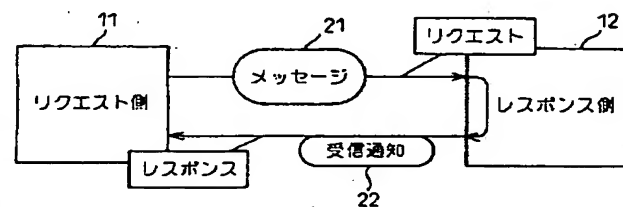


図2 リクエスト側からレスポンス側にメッセージを送信する場合



【図6】

【図3】

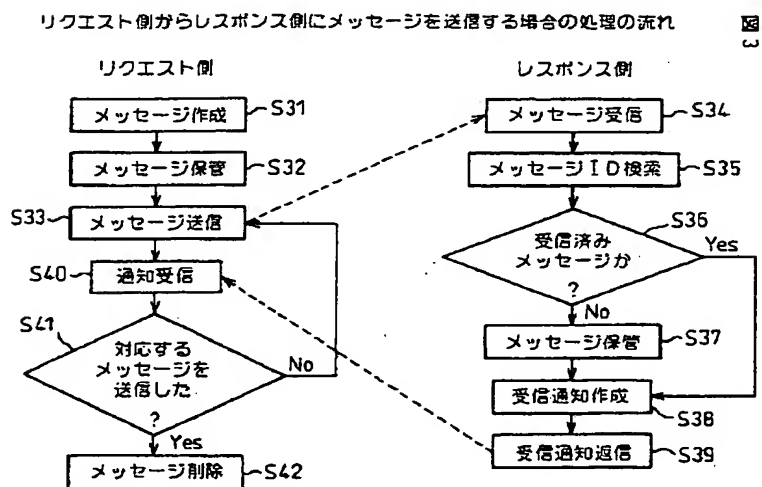


図6 メッセージIDを埋め込んだメッセージの例

```
<Order xmlns="http://jp.fujitsu.com/order">
  <messageId>12345678</messageId>
  <userInfo>
    :
  </userInfo>
</Order>
```

【図7】

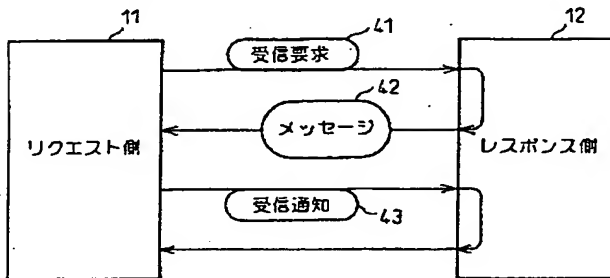
図7 ネームスペースを使ってメッセージIDを埋め込んだメッセージの例

```
<Order ssrs:messageId="12345678"
  xmlns="http://jp.fujitsu.com/order"
  xmlns:ssrs="http://jp.fujitsu.com/ssrs">
  <userInfo>
    :
  </userInfo>
</Order>
```

【図4】

図4

レスポンス側からリクエスト側にメッセージを送信する場合



【図8】

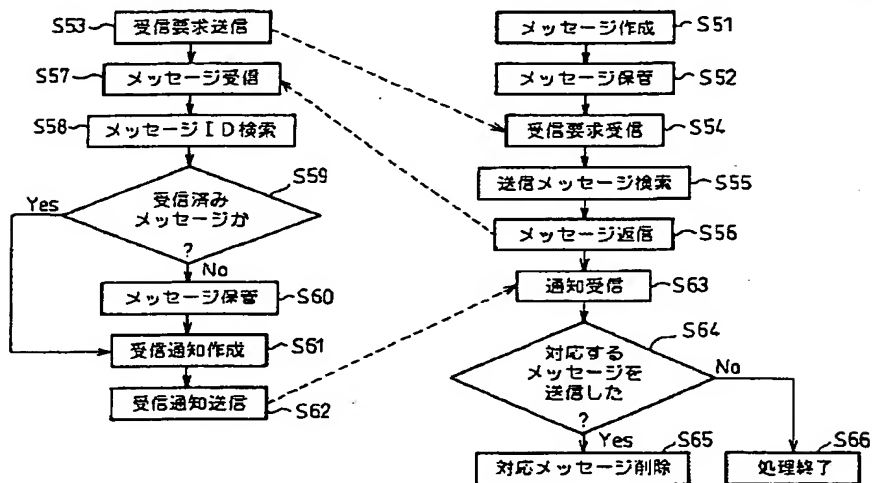
図8 任意のフィールドで一貫性を保証するメッセージの例

```

<Order xmlns="http://jp.fujitsu.com/order">
  <orderId>order01234567</orderId>
  <userInfo>
    :
  </userInfo>
</Order>
  
```

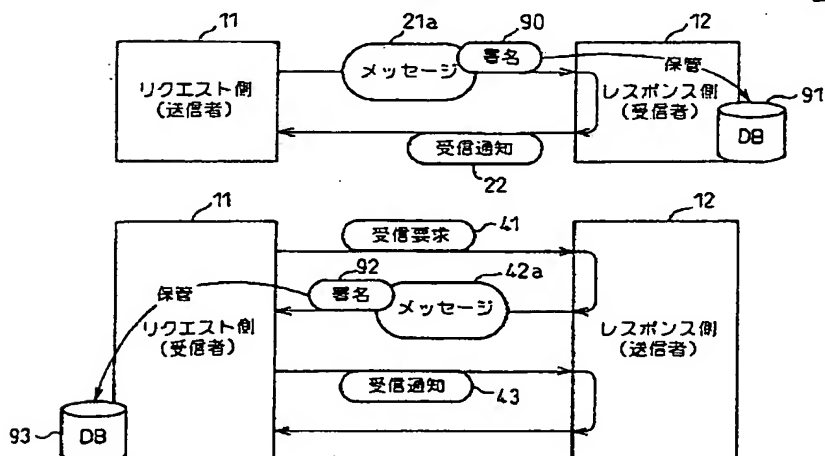
【図5】

レスポンス側からリクエスト側にメッセージを送信する場合の処理の流れ

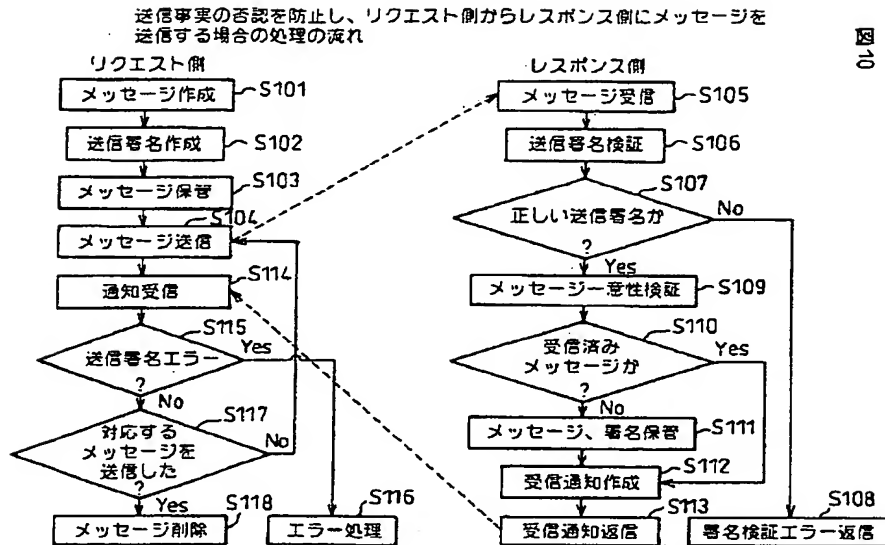


【図9】

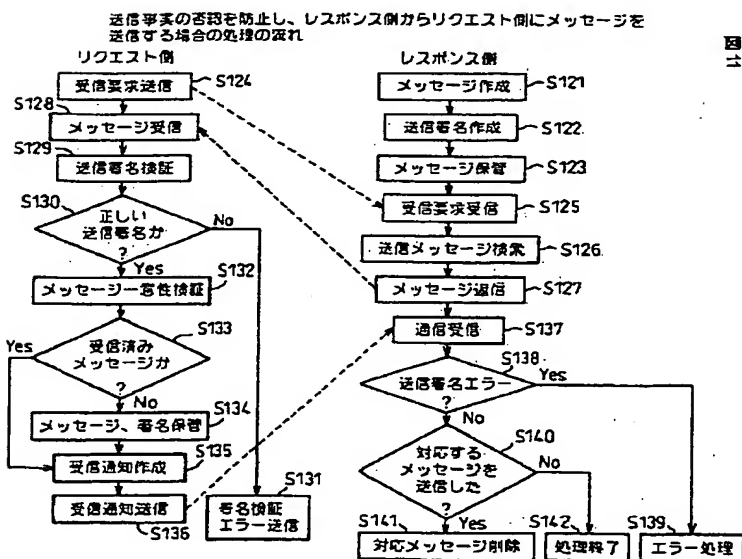
送信者による送信事実の否認防止



【図10】

図
10

【図11】



【図25】

図
25

図 25

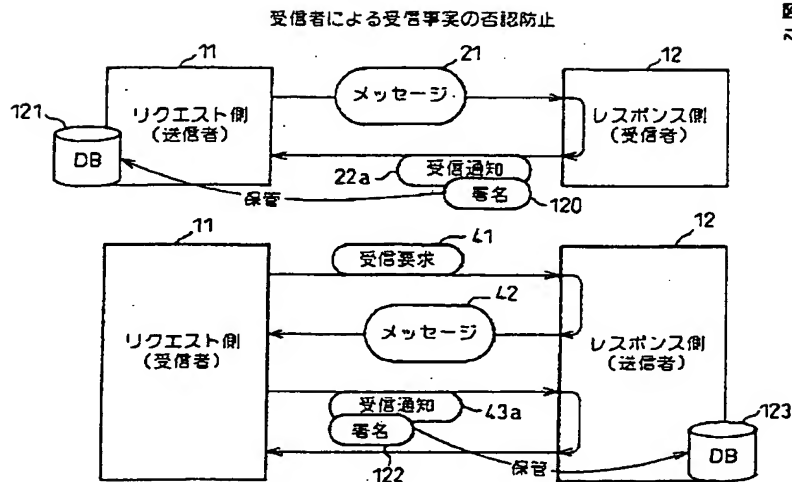
署名有り送信メッセージ (3)

```

</ds:SignedInfo>
<ds:SignatureValue>
  BL8jdfToEb1I/vXcMZNNjPOV...
</ds:SignatureValue>
<ds:KeyInfo>
  <wsse:SecurityTokenReference>
    <wsse:Reference URI="#X509Token"/>
  </wsse:SecurityTokenReference>
</ds:KeyInfo>
</ds:Signature>
</wsse:Security>
</SOAP:Header>
<SOAP:Body>
  <gip:GetItemPrice xmlns:gip="Some-URI">
    <gip:itemnumber>product12345</gip:itemnumber>
    </gip:GetItemPrice>
  </SOAP:Body>
</SOAP:Envelope>

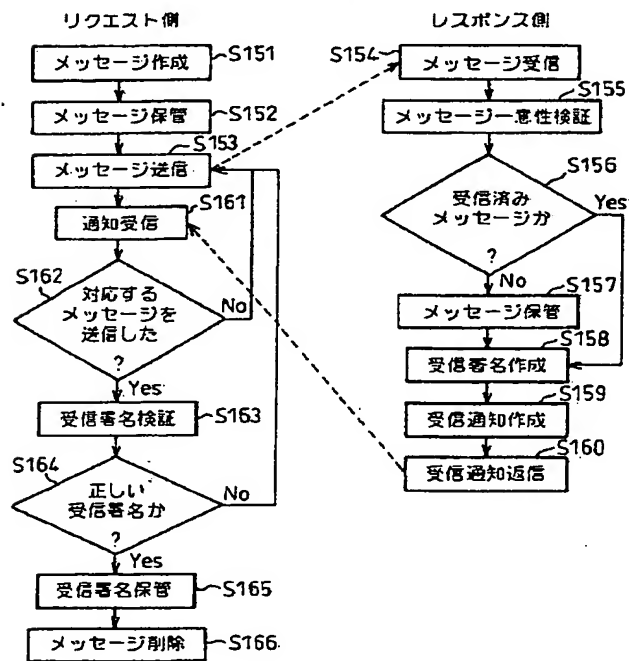
```


【図12】



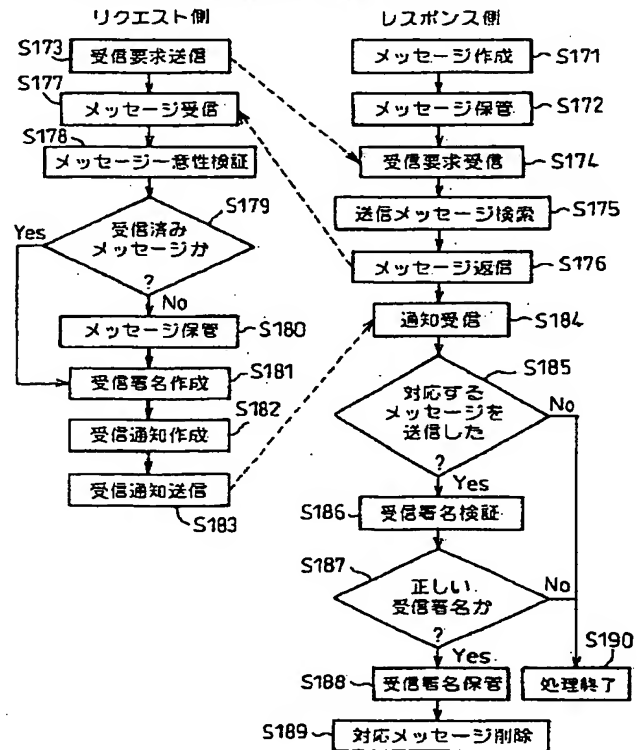
【図13】

図13
受信事実の否認を防止し、リクエスト側からレスポンス側にメッセージを送信する場合の処理の流れ

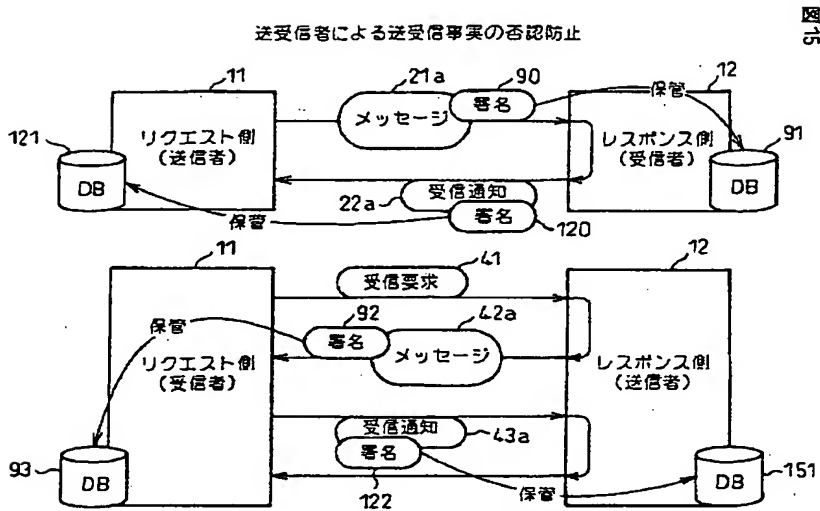


【図14】

図14
受信事実の否認を防止し、レスポンス側からリクエスト側にメッセージを送信する場合の処理の流れ

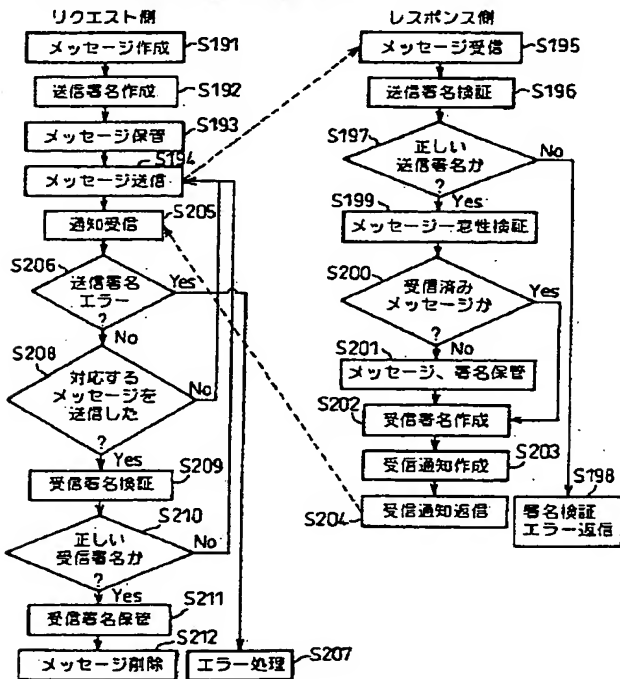


【図15】



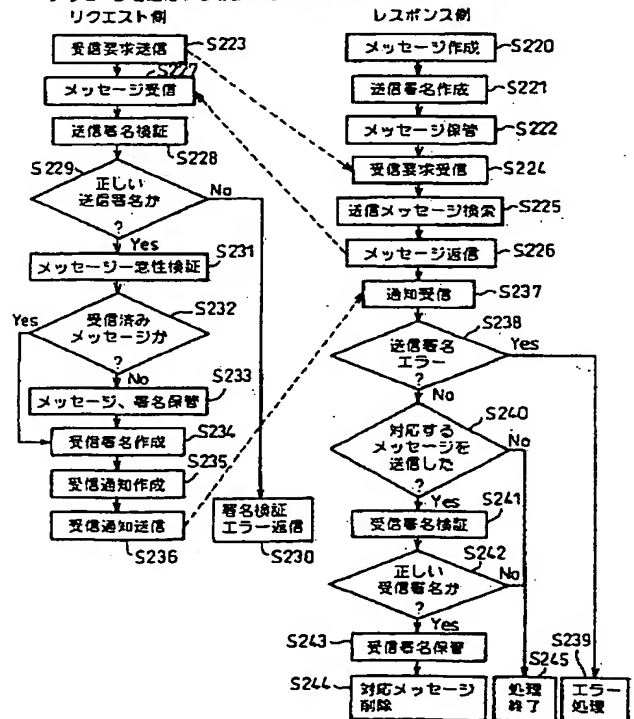
【図16】

図16
送受信事実の否認を防止し、リクエスト側からレスポンス側に
メッセージを送信する場合の処理の流れ



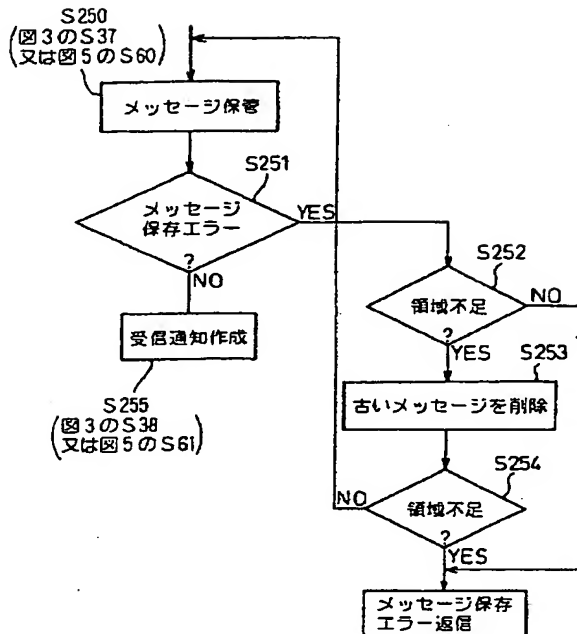
【図17】

図17
送受信事実の否認を防止し、レスポンス側からリクエスト側に
メッセージを送信する場合の処理の流れ



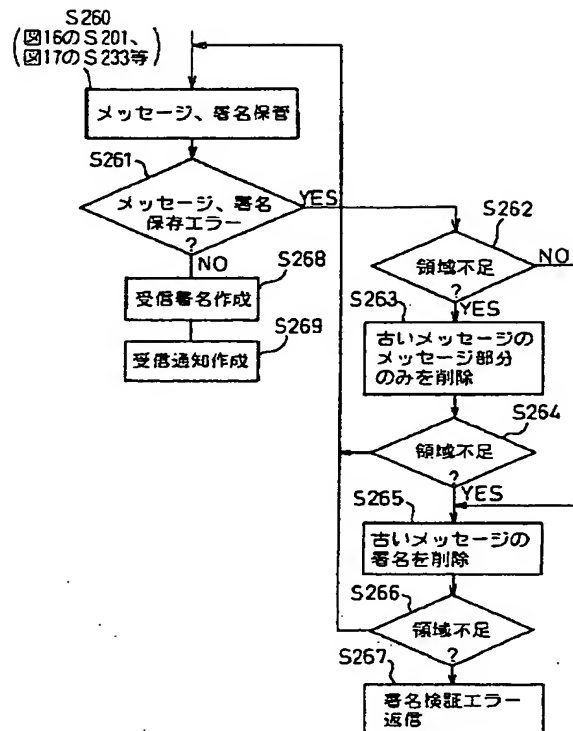
【図 18】

図 18



【図 19】

図 19



【図 20】

メッセージの送信要求

```

<SOAP:Envelope
  xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP:Header>
    <rm:MessageHeader xmlns:rm="http://schemas.xmlsoap.org/rm"
      SOAP:mustUnderstand="1">
      <rm:From>requester@anyuri.com</rm:From>
      <rm:To>responder@someuri.com</rm:To>
      <rm:Service>urn:services:ItemQuoteService</rm:Service>
      <rm:MessageType>Request</rm:MessageType>
    </rm:MessageHeader>
  </SOAP:Header>
  <SOAP:Body>
  </SOAP:Body>
</SOAP:Envelope>
  
```

【図 28】

図 28

署名有り受信確認メッセージ (3)

```

</ds:Transforms>
<ds:DigestMethod Algorithm=
  "http://www.w3.org/2000/09/xmldsig#sha1"/>
<ds:DigestValue>EULddytSo1...</ds:DigestValue>
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>
  BL8jdToEb1l/vXcMZNNjPOV...
</ds:SignatureValue>
<ds:KeyInfo>
  <wsse:SecurityTokenReference>
    <wsse:Reference URI="#X509Token"/>
  </wsse:SecurityTokenReference>
  </ds:KeyInfo>
</ds:Signature>
</wsse:Security>
</SOAP:Header>
<SOAP:Body>
</SOAP:Body>
</SOAP:Envelope>
  
```

【図 2 1】

署名無し送信メッセージ

図 21

```

<SOAP:Envelope
  xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP:Header>
    <rm:MessageHeader xmlns:rm="http://schemas.xmlsoap.org/rm"
      SOAP:mustUnderstand="1">
      <rm:From>requester@anyuri.com</rm:From>
      <rm:To>responder@someuri.com</rm:To>
      <rm:Service>urn:services:itemQuoteService</rm:Service>
      <rm:MessageType>Message</rm:MessageType>
      <rm:MessageId>20020907-045261-0450@anyuri.com</rm:MessageId>
      <rm:Timestamp>2002-09-07T10:19:07</rm:Timestamp>
    </rm:MessageHeader>
    <rm:ReliableMessage xmlns:rm="http://schemas.xmlsoap.org/rm"
      SOAP:mustUnderstand="1">
      <rm:AckRequested SOAP:mustunderstand="1" rm:version="1.0"
        rm:signed="false" />
      <rm:DuplicateElimination />
    </rm:ReliableMessage>
  </SOAP:Header>
  <SOAP:Body>
    <gip:GetItemPrice xmlns:gip="Some-URI">
      <gip:itemnumber>product12345</gip:itemnumber>
    </gip:GetItemPrice>
  </SOAP:Body>
</SOAP:Envelope>

```

【図 2 2】

署名無し受信確認メッセージ

図 22

```

<SOAP:Envelope
  xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP:Header>
    <rm:MessageHeader xmlns:rm="http://schemas.xmlsoap.org/rm"
      SOAP:mustUnderstand="1">
      <rm:From>responder@someuri.com</rm:From>
      <rm:To>requester@anyuri.com</rm:To>
      <rm:Service>urn:services:itemFilingService</rm:Service>
      <rm:MessageType>Acknowledgment</rm:MessageType>
      <rm:MessageId>20020907-045261-0450@anyuri.com</rm:MessageId>
      <rm:Timestamp>2002-09-07T10:19:07</rm:Timestamp>
      <rm:RefToMessageId>20020907-44-070@uri.com</rm:RefToMessageId>
    </rm:MessageHeader>
    <rm:ReliableMessage xmlns:rm="http://schemas.xmlsoap.org/rm"
      SOAP:mustUnderstand="1">
      <rm:Acknowledgment SOAP:mustunderstand="1" rm:version="1.0" />
    </rm:ReliableMessage>
  </SOAP:Header>
  <SOAP:Body>
  </SOAP:Body>
</SOAP:Envelope>

```

【図 2 3】

署名有り送信メッセージ (1)

図 23

```

<SOAP:Envelope
  xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  <SOAP:Header>
    <rm:MessageHeader xmlns:rm="http://schemas.xmlsoap.org/rm"
      SOAP:mustUnderstand="1">
      <rm:From>requester@anyuri.com</rm:From>
      <rm:To>responder@someuri.com</rm:To>
      <rm:Service>urn:services:itemQuoteService</rm:Service>
      <rm:MessageType>Message</rm:MessageType>
      <rm:MessageId>20020907-045261-0450@anyuri.com</rm:MessageId>
      <rm:Timestamp>2002-09-07T10:19:07</rm:Timestamp>
    </rm:MessageHeader>
    <rm:ReliableMessage xmlns:rm="http://schemas.xmlsoap.org/rm"
      SOAP:mustUnderstand="1">
      <rm:AckRequested SOAP:mustUnderstand="1" rm:version="1.0"
        rm:signed="false"/>
      <rm:DuplicateElimination/>
    </rm:ReliableMessage>
    <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/04/secext">
      <wsse:BinarySecurityToken
        Value Type="wsse:X509v3"
        Encoding Type="wsse:Base64Binary"
        Id="X509Token">
        MIIeZzCCA9CgAwIBAgIQEmtJZc0qrKh5i...

```

【図 2 4】

署名有り送信メッセージ (2)

図 24

```

</wsse:BinarySecurityToken>
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <ds:SignedInfo>
    <ds:CanonicalizationMethod Algorithm=
      "http://www.w3.org/2001/10/xml-exc-c14n#" />
    <ds:SignatureMethod Algorithm=
      "http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
    <ds:Reference URI="">
      <ds:Transforms>
        <ds:Transform Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
          <ds:XPath> ancestor-or-self::rm:MessageHeader or ancestor-or-self::rm:Reliable
            Message or ancestor::SOAP:Body</ds:XPath>
        </ds:Transform>
        <ds:Transform Algorithm=
          "http://www.w3.org/2001/10/xml-exc-c14n#" />
        </ds:Transforms>
        <ds:DigestMethod Algorithm=
          "http://www.w3.org/2000/09/xmldsig#sha1" />
        <ds:DigestValue>EULddytSo1...</ds:DigestValue>
      </ds:Reference>

```

【図 26】

署名有り受信確認メッセージ (1)

図 26

```

<SOAP:Envelope
  xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP:Header>
    <rm:MessageHeader xmlns:rm="http://schemas.xmlsoap.org/rm"
      SOAP:mustUnderstand="1">
      <rm:From>responder@someuri.com</rm:From>
      <rm:To>requester@anyuri.com</rm:To>
      <rm:Service>urn:schemas:ItemFilingService</rm:Service>
      <rm:MessageType>Acknowledgment</rm:MessageType>
      <rm:MessageId>20020907-045261-0450@anyuri.com</rm:MessageId>
      <rm:Timestamp>2002-09-07T10:19:07</rm:Timestamp>
      <rm:RefToMessageId>20020907-44-070@uri.com</rm:RefToMessageId>
    </rm:MessageHeader>
    <rm:ReliableMessage xmlns:rm="http://schemas.xmlsoap.org/rm"
      SOAP:mustUnderstand="1">
      <rm:Acknowledgment SOAP:mustUnderstand="1" rm:version="1.0" />
    </rm:ReliableMessage>
    <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/04/secext">
      <wsse:BinarySecurityToken
        Value Type="wsse:X509v3"
        Encoding Type="wsse:Base64Binary"
        Id="X509Token">

```

【図 27】

署名有り受信確認メッセージ (2)

図 27

```

MUEZzCCA9CgAwIBAgIQEmtJZc0rqKh5i...
</wsse:BinarySecurityToken>
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <ds:SignedInfo>
    <ds:CanonicalizationMethod Algorithm=
      "http://www.w3.org/2001/10/xml-exc-c14n#" />
    <ds:SignatureMethod Algorithm=
      "http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
    <ds:Reference URI="">
      <ds:Transforms>
        <ds:Transform Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
          <ds:XPath>ancestor-or-self::rm:MessageHeader or ancestor-or-self::rm:Reliable
            Message </ds:XPath>
        </ds:Transform>
        <ds:Transform Algorithm=
          "http://www.w3.org/2001/10/xml-exc-c14n#" />
        </ds:Transforms>
      <ds:DigestMethod Algorithm=
        "http://www.w3.org/2000/09/xmldsig#sha1" />
      <ds:DigestValue>EULd0ytSo1...</ds:DigestValue>
    </ds:Reference>
    <ds:Reference URI="rm:20020907-045261-0450@anyuri.com">
      <ds:Transforms>
        <ds:Transform Algorithm=
          "http://www.w3.org/2001/10/xml-exc-c14n#" />

```

フロントページの続き

Fターム(参考) 5K018 BA03 FA02
 5K034 AA06 DD01 EE10 HH01 HH02
 HH11 MM05